Computer Science Department

TECHNICAL REPORT

Inductive Inference and Encoding

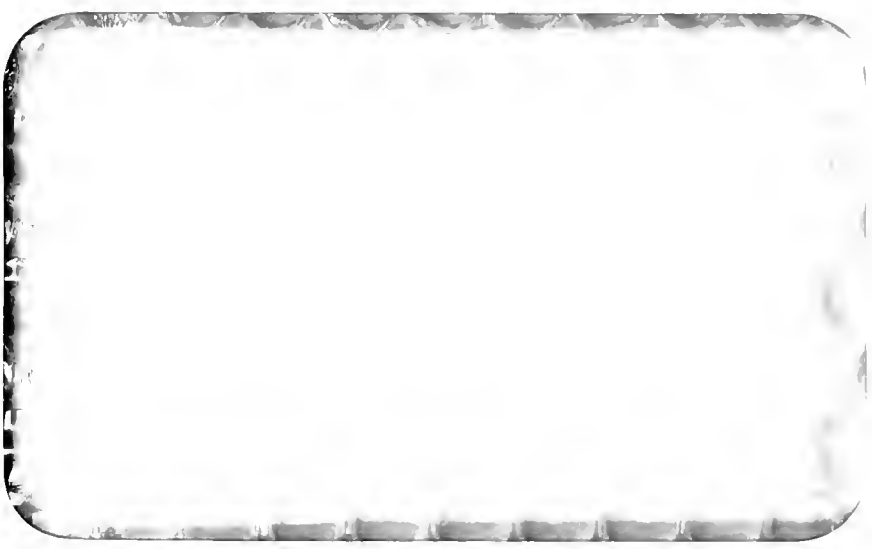
Pasquale Caianiello

Technical Report 372

May 1988

NEW YORK UNIVERSITY

Department of Computer Science
Courant Institute of Mathematical Sciences
251 MERCER STREET, NEW YORK, NY 10012



Inductive Inference and Encoding

Pasquale Caianiello

Technical Report 372

May 1988

Inductive Inference and Encoding

Pasquale Caianiello
Courant Institute, New York University
251 Mercer Street, NY, NY 10012, USA
(caianiel@nyu-csd2.arpa)

April 7, 1988

Abstract

The theory of Inductive Inference studies the problems related to finding structural descriptions from samples. In the present paper we suggest that this process can be viewed as the result of consecutive compressions of the data into more efficient representations.

We provide two ways to improve the encoding of an abstract text and we show how they could constitute the base of a learning procedure. We test the ideas on a simple example.

1 Introduction

One of the recognized problems in the theory of inductive inference is that of *comparing hypotheses* [Angluin and Smith 1983, Section 5], which is to find criteria and quantitative measures to evaluate the fitness and the simplicity of an explanation. The perfect candidate as a measure in an algorithmic setting is, of course, the distance from Kolmogorov complexity, K. In this way it is possible to derive a formal conceptualization of the *Occam razor* [Koppel 1987]. Unfortunately, only its intuitive significance can be used in actual applications. Its definition is nonconstructive and Kolmogorov himself proved that K is not a partial recursive function [Zvonkin and Levine 1970, Theorem 1.5]. Nonetheless the theoretical importance of this approach is fundamental, and one is advised to look for a more constructive counterpart, a measure of complexity based on what can be observed of a phenomenon.

The use of Shannon entropy, H, as a grammatical complexity measure has been attempted [Cook et al. 1976] although only in the more natural scenery of stochastic grammars. In this paper we propose a more systematic

use of it, stimulated by results which exhibit H's profound relations with Kolmogorov complexity [Zvonkin and Levine 1970, Chaitin 1975, Leung-Yang-Cheong and Cover 1978, Cover 1985]. We will be able to establish an operative link between Shannon entropy and Kolmogorov complexity utilizing an elaboration on the concept of *minimal description* due to [Koppel 1987]: A description of a string $\mathcal{S}$ is a couple (P, D), where P is a program and D is the data which when given as input to an Universal Turing Machine would output $\mathcal{S}$. A description is minimal if $|P|+|D|$ is minimal. Intuitively, P is a description of the string structure and D specifies that string among the ones sharing the same structure. In our information-theoretical setting P will represent the code used and D the encoding of S under the code P. A Universal Interpreter will reconstruct S given (P, D).

Section 2 will describe two ways for improving the encoding of a text, giving the condition under which they yield a gain and a quantitative estimate of that gain.

In Section 3 the results obtained will be applied to the theory of inductive inference. Current theoretical approaches rely on an enumeration of the possible descriptions (usually grammars) according to an order imposed on them by a complexity measure. The discussion of Section 2 will point out a way, using Shannon entropy as complexity measure, to reverse the enumeration, allowing to look at the inductive process as one which proceeds towards descriptions with lower and lower complexity.

Section 4 will be devoted to a discussion of the applicative work, which is under way, that this conceptualization allows.

2 Encoding a Text

Definitions and Notational Conventions

- $\mathcal{T}$ a two-way infinite text, $\mathcal{T}_N$ a substring of length N.
- $A = \{a_i\}_{i=1}^{n}$ the alphabet of the text
- $S = \{s_j\}_{j=1}^{m}$ a *syllabary* for $\mathcal{T}$, that is a set of strings of symbols in A such that $\mathcal{T} = ...s_{i_{-1}}s_{i_0}s_{i_1}...$, $\mathcal{T}$ can be viewed as a text on the alphabet S in a unique way, S is a *uniquely decipherable* code for T. In particular A is a syllabary for $\mathcal{T}$.
- $C = \{c_i\}_{i=1}^{n'}$ a *classification*, i.e. a partition of A, hence $\sum_{i=1}^{n'} |c_i| = n$

- $\overline{S}^C = \{\overline{s}_j\}_{j=1}^{m'}$ the *reduced syllabary* under the classification C. To each $s \epsilon S$, $s = a_{i_1} ... a_{i_{l(s)}}$ corresponds an $\overline{s} \epsilon \overline{S}^C$, $\overline{s} = c_{j_1} ... c_{j_{l(\overline{s})}}$ where c_{j_h} is the class of a_{i_h}. Each element $\overline{s} \epsilon \overline{S}^C$ is a class of syllables of S.
- $l_S(s)$ the length of the string s over the alphabet S. If S is understood, the subscript can be omitted
- $p_S(s)$ the probability of occurrence of the syllable $s \epsilon S$ in the text $\mathcal{T}$, that is, its relative frequency with respect to the syllabary S. The subscript is omitted when there is no ambiguity about the syllabary that should be considered.
- $p(\_)$ the probability of the event in the argument.
- log x the logarithm in base 2 of x

Example 1 Consider the language $L = ((ab)^* \cup (cd)^*) \cdot (\epsilon\epsilon)^*$ and let T be the a possible sequence of justappositions of words in L. Then $A = \{a, b, c\}$ is the alphabet of T, $S = \{ab, cd, \epsilon\epsilon\}$ is a syllabary for T, $C = \{c_1, c_2, c_3\}$ where $c_1 = \{a, c\}, c_2 = \{b, d\}, c_3 = \{e\}$ is a classification and the associated reduced syllabary is $\overline{S}^C = \{c_1c_2, c_3c_3\}$. The original language could be expressed as $L = (c_1c_2)^* \cdot (c_3c_3)^*$.

Shannon Entropy

We are presented with an infinite text $\mathcal{T}$ and the problem of finding the optimal encoding in bits. If $\mathcal{T}$ is encoded considering only the lowest level alphabet A, the lower bound for average length per character of an optimal uniquely decipherable encoding $P(A)$ is

$$H(A) = -\sum_{i=1}^{n} p(a_i) \log p(a_i)$$

A substring $\mathcal{T}_N$ of length N of $\mathcal{T}$ would have an encryption D(A) with an average length in bits

$$|D(A)| = N \cdot H(A) = -N \sum_{i=1}^{n} p(a_i) \log p(a_i) + |P(A)|$$

where $|P(A)|$ is intended as the length of the code table for the symbols.

Encoding through a Syllabary

We want now to investigate under which conditions an encoding done under a syllabary is convenient with respect to the one done under the low level alphabet.

Let S be a syllabary for $\mathcal{T}$. We have that:
The expected length of a syllable in S is

$$\langle |s| \rangle = \sum_{j=1}^{m} p(s_j) l(s_j)$$

The expected number of syllables in $\mathcal{T}_N$ is

$$N' = \frac{N}{\sum_{j=1}^{m} p(s_j) l(s_j)}$$

The expected length of a syllable of an optimal encoding $P(S)$

$$H(S) = -\sum_{j=1}^{m} p(s_j) \log p(s_j)$$

Under the syllabary S, the expected number of bits in which $\mathcal{T}_N$ would be encoded is

$$|D(S)| = N' \cdot H(S) + |P(S)| = -\frac{N \sum_{j=1}^{m} p(s_j) \log p(s_j)}{\sum_{j=1}^{m} p(s_j) l(s_j)} + |P(S)| \qquad (1)$$

Let us define

$$\lambda_S = \frac{\sum_{j=1}^{m} p(s_j) \log p(s_j)}{\sum_{j=1}^{m} p(s_j) l(s_j)}$$

For large N the term $|P(S)|$ can be ignored, hence λ_S can be thought as the compressing capacity of the syllabary S.

The gain in compression obtained under the encoding S is

$$\frac{\lambda_S}{\lambda_A} = \lim_{N \to \infty} \frac{|D(S)|}{|D(A)|} = \frac{\sum_{j=1}^{m} p(s_j) \log p(s_j)}{\sum_{j=1}^{m} p(s_j) l(s_j) \sum_{i=1}^{n} p(a_i) \log p(a_i)}$$

The encoding under the syllabary S is convenient whenever $\lambda_S / \lambda_A < 1$ and it is optimal when λ_S is minimal. (If we are dealing with a finite text then the limit operation will not apply and we will have to consider also the length of the syllable table $P(S)$).

Example 2 Let us assume that $p(a) = p(a')\ \forall a, a' \epsilon A$ and $p(s) = p(s')$ $\forall s, s' \epsilon S$. From our point of view, this is the situation when no information about the frequency of occurrence of alphabet symbols and syllables is available or, equivalently, when in a particular application its retrieval is not advised. It is well known that this assumption would maximize the entropy function and then the average length per character of the code.

Let us also assume that $s = \sum_{j=1}^{m} p(s_j) l(s_j)$ is the average length of a syllable in S. Our expression for λ_S becomes

$$\lambda_S = \frac{\log m}{s}$$

Under this assumptions it is convenient to use the syllabary whenever $n^s > m$.

Example 3 Let $\mathcal{T}$ a periodic text, that is the repetition of the same string s. Then there a syllabary, namely $S = \{s\}$, which would make the compressing factor $\lambda_S = 0$. The gain obtained through S would be, in fact, infinite, we would be able to give a finite description of an infinite text.

Encoding through a Classification

A similar line of thought will now guide us in discovering that in encoding a text $\mathcal{T}$ there can be a gain in classifying the symbols of an alphabet taking in account the syllabic structure of $\mathcal{T}$. Under the classification C we have: The average length of the syllables in S and the average length $\langle |s| \rangle$ of the reduced syllables in $\overline{S}^C$ is the same,

$$\langle |s| \rangle = \sum_{j=1}^{m} p(s_j) l_A(s_j) = \sum_{j=1}^{m'} p(\overline{s}_j) l_{\overline{S}^C}(\overline{s}_j)$$

As a consequence, the average number N' of syllables in S and of the reduced syllables in $\overline{S}^C$ of a substring $\mathcal{T}_{\mathcal{N}'}$ of length N of $\mathcal{T}$ is the same. The average length of the code for a reduced syllable is

$$H(\overline{S}) = -\sum_{j=1}^{m'} p(\overline{s}_j) \log p(\overline{s}_j)$$

The average information required to determine a syllable s out of its class $\overline{s}$ is of $H(S, C)$ bits, where

$$H(S, C) = -\sum_{j=1}^{m'} p(\overline{s}_j) \sum_{h=1}^{l(\overline{s}_j)} \sum_{i=1}^{|c_{j_h}|} p(c_{j_h} = a_{j_i}) \log p(c_{j_h} = a_{j_i})$$

Let $P(C)$ the table for the classification C, then under the induced classification $\overline{S}^C$ the average length in bits of the codification of $\mathcal{T}_N$ would be

$$\left|D(\overline{S})\right| = N' \cdot (H(\overline{S}) + H(S,C)) + |P(C)|$$

The gain of the induced classification over the syllabary S is

$$\overline{\lambda}_S^C = \lim_{N\to\infty} \frac{\left|D(\overline{S})\right|}{|D(S)|}$$

hence

$$\overline{\lambda}_S^C = \frac{\sum_{j=1}^{m'} p(\overline{s}_j)\log p(\overline{s}_j) + \sum_{j=1}^{m'} p(\overline{s}_j)\sum_{h=1}^{l(\overline{s}_j)}\sum_{i=1}^{|c_{j_h}|} p(c_{j_h}=a_i)\log p(c_{j_h}=a_i)}{\sum_{j=1}^{m} p(s_j)\log p(s_j)}$$

As before there is an effective gain only when $\overline{\lambda}_S^C < 1$ and the maximum gain is reached when $\overline{\lambda}_S^C$ is minimal.

The overall gain of the encoding by means of the syllabary S and the classification C is

$$\lambda_{S,C} = \overline{\lambda}_S^C \cdot \lambda_S$$

It can happen that $\lambda_{S,C} < 1$ even if $\lambda_S/\lambda_A > 1$. The right classification can increase the value of an otherwise useless syllabary.

Example 4 Let c be the average number of elements in the classes of C and s the average length of a syllable. As in Example 2, let us assume that no information about probability is available. Then

$$\overline{\lambda}_S^C = \frac{\log m' + s\log c}{\log m}$$

and

$$\lambda_{S,C} = \frac{\log m' + s\log c}{s\log n}$$

A classification is convenient when $m' < \frac{m}{c^s}$. The combination of a syllabary and a classification is convenient when $m' < n^{'s}$.

3 Inductive Inference

We have seen that in some situation a shift in point of view can lower the length of the encoding of a finite substring of $\mathcal{T}$ or, in other terms, can lower its complexity (better *perceived* complexity). We can, view a learning procedure as any one which operates such shifts.

In the previous section we have given two ways to produce a change in point of view. We like to parallel them to the two processes of analysis, in which the right code words constituting a text are discovered, and of synthesis in which the right classification of those symbols is found (*right*, in the light of what observed in section 2, stands for *most convenient*).

We can now outline the procedure which immediately follows our considerations:

Algorithm 1

1 Set $i = 0$ and $(S, C) = (A, A)$

2 Set $(S_i, C_i) = (S, C)$; $i = i + 1$

3 Choose (S, C), where S is a syllabary, $|s| < l, \forall s \epsilon S$, and C a classification, such that $\lambda_{S,C}$ is minimal for $\mathcal{T}$ viewed as a text on the alphabet $\overline{S}_{i-1}^{C_{i-1}}$

4 If $\lambda_{S,C} \geq 1$ then stop

5 Go to step 2

This algorithm will construct a hierarchy (S_i, C_i) with possibly infinite levels, which constitutes the structure of the text. We need to enforce the artificial limiting condition on the maximum size of the syllables sought for, to insure the termination of Step 3. Different values of l might lead to very different results. For any value of l there exists a structured text whose structure cannot be discovered by Algorithm 1 limited by l.

It is obvious that in actual applications of Algorithm 1, one need not push the choice of Step 3 to the extreme optimality. One could as well substitute Step 3 and 4 with

3′ Choose (S, C), where S is a syllabary and C a classification, such that $\lambda_{S,C}/\lambda_{S_{i-1},C_{i-1}} < 1$ if it exists, else stop.

In that situation the procedure will still "learn" something, and its effectiveness will be related to how close the choice in Step 3′ is to the optimal. In this way we could widen the actual applicability of Algorithm 1, originally restricted by the high complexity of Step 3. We could, in fact, use more illuminated procedures than the dull thorough search over all the possible syllabaries. Extensive work, from which the approach of the present paper stems, has already been done to tackle this problem [Caianiello and Capocelli 1971, 1976]. If one limits the search only to uniquely decipherable codes the task is considerably easier. [Caianiello and Capocelli 1976] give four different algorithm, each one seeking for a different type of code. All of the algorithms show *adaptive* characteristics, they *identify in the limit* [Gold 1967] the structure sought for.

The search for comparably better classification procedures is currently under way.

4 Experimental Work and Applications: Grammatical Inference

The procedure outlined in Section 3 by Algorithm 1 provides a constructive method to achieve grammatical inference which, of course, needs to be refined according to the needs of the specific applications. However, there is still the need for much work in order to formalize all the necessary steps towards the formulation of a solid general project.

Before undertaking such an effort one might feel the necessity to experiment the validity of the ideas involved. The first kind of experiment that one can think of is to check how other existing approaches fit in the scheme proposed. Let us consider a text $\mathcal{T}$ consisting of set of 100 words randomly generated using the following grammar, a simplified version of the one given by [Grishman 1986, Section 2.4.1].

<S> ::= <SUBJ> <VERB> <OBJ>
<SUBJ> ::= <NSTG>
<PN> ::= P <NSTG>
<NSTG> ::= <LNR>
<LNR> ::= <LN> N <RN>
<LN> ::= <TPOS> <APOS>
<TPOS> ::= T | *null*
<APOS> ::= ADJ | *null*
<RN> ::= <PN> | *null*

<VERB> ::= <LTVR>
<LTVR> ::= <LV> TV <RV>
<LV> ::= D | $null$
<RV> ::= D | <PN> | $null$
<LVR> ::= <LV> V <RV>
<OBJ> ::= <NSTG> | <TOVO> | $null$
<TOVO> ::= to <LVR> <OBJ>

If we run twice the algorithm N in [Caianiello and Capocelli 1976] on such a set of sentences (which corresponds to two iterations of Steps 1,2 and 3 of Algorithm 1, hence to finding the second level structure), we get the syllabary

$$S = \left\{ \begin{array}{cccc} (to\ V) & (P\ T\ N) & (T\ ADJ\ N) & (P\ N) \\ (P\ ADJ\ N) & (to\ D\ V) & (ADJ\ N) & (D) \\ (N) & (T\ N) & (P\ T\ ADJ\ N) & (TV) \end{array} \right\}$$

In the following table we can compare the values for the quantities H $\langle |s| \rangle$ and λ for the alphabet A of the terminal symbols of the grammar, S, and S_2 consisting of all the digrams appearing in the text and all the sentence-ending monograms (to guarantee parsing of sentences with an odd number of components).

	H	$\langle \|s\| \rangle$	λ
A	2.701	1	2.701
S	3.387	2.02	1.676
S_2	3.254	1.49	2.183

As we see there is a sensible difference in the gain obtained using algorithm N. Moreover, if we notice that the grammar chosen is one which describes (even if incompletely) the word classes phrase structure of English, we find that the syllables discovered by algorithm N coincide with the ones which a person (not only a linguist) would normally consider to be the elementary strings of the phrase structure. The results obtained by [Caianiello and Capocelli 1971, 1976] in discovering syllables in Italian texts and the one here exposed allows some conjectures which will guide our future work: λ

reaches a local minimum over syllabaries which could be and have been discovered by means of other "reasonable" considerations. The process which guides language use is the same at different levels and linguistic ability is the superimposition of the same operations on different substrata.

Acknowledgements

I received much help and advice in the development of these ideas from Ernest Davis and Bhubaneswar Mishra, whom I wish to thank.

References

Angluin and Smith 1983 D. Angluin and C.H.Smith, Inductive Inference: Theory and Methods, *Computing Surveys* 15 (1983), 237-268

Caianiello and Capocelli 1971 E.R. Caianiello and R. Capocelli, On form and language: The Procustes Algorithm for feature extraction, *Kibernetik* 8 (1971), 223-233.

Caianiello and Capocelli 1976 E.R. Caianiello and R. Capocelli, Structural analysis of hierarchical systems, *Proc. 3rd Joint Conf. Pattern Recognition* (1976)

Chaitin 1975 G.H. Chaitin, A theory of program size complexity formally equivalent to information theory, *Jour. of the ACM* 22 (1971), 329-340.

Cook et al. 1976 C.M.Cook, A. Rosenfield, and A.R. Aronson, Grammatical inference by hill climing, *Information Sciences* 10 (1976), 59-80

Cover 1985 T. Cover, Kolmogorov complexity, data compressing, and inference, in *The Impact of Processing Techniques on Communications*, J.K.Skwirzynski ed., Martinus Nijhoff Publisher (1985)

Gold 1967 E.M. Gold, Language identification in the limit, *Information and Control* 10 (1967), 447-474

Grishman 1986 R. Grishman, *Computational Linguistics: an Introduction*, Cambridge University Press (1986)

Koppel 1987 M. Koppel, Structure, *manuscript* (1987)

Leung-Yang-Cheong and Cover 1978 S.K. Leung-Yang-Cheong and T. Cover, Some equivalences between Shannon entropy and Kolmogorov complexity, *IEEE Trans. on Information Theory* 24 (1978), 331-337

Zvonkin and Levin 1970 A.K. Zvonkin and L.A. Levin,The complexity of finite objects and the development of the concepts of information and randomness by means of the theory of algorithms, *Russian Mathematical Surv.* 25 (1970), 83-124

NYU COMPSCI TR-372 c.1
Caianiello, Pasquale
Inductive inference and
encoding.